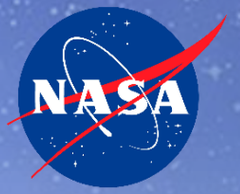
January, 2022



Parallelized convolutional interleaver implementation for efficient DDR memory access

Jennifer N. Downey, Mary Jo W. Shalkhauser, and Thomas P. Bizon

NASA Glenn Research Center

Cleveland, Ohio



Introduction

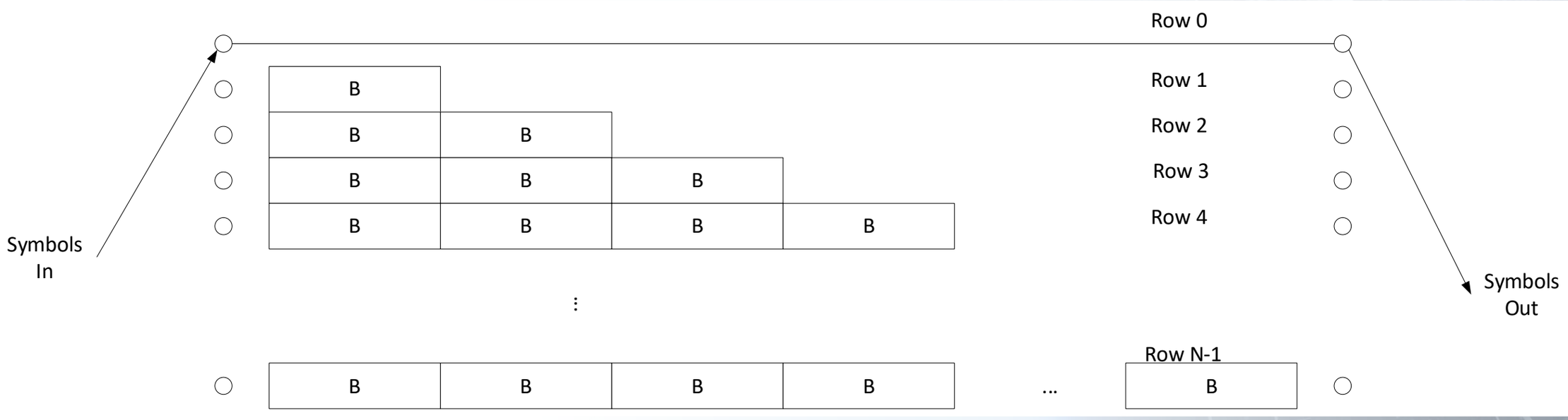
- NASA is using the CCSDS Optical Communications High Photon Efficiency (HPE) waveform on future missions: Optical Artemis-2 Orion (O2O), Psyche
- A convolutional interleaver is used in of the CCSDS HPE standard to correct for burst errors
- A previous convolutional interleaver implementation in FPGA block RAM exceeded the memory elements for large interleaver sizes

 → A DDR implementation is necessary to implement sizes required for O2O
- The DDR interface is 512 bits, but symbols are represented by 8 bits in the interleaver

 → A parallel implementation is required

Background

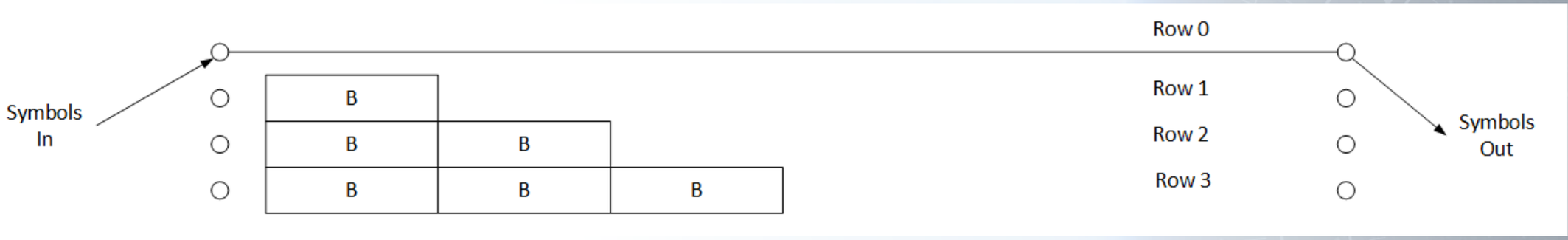
A convolutional interleaver can be implemented with a shift register approach in block RAM in the FPGA.



Background

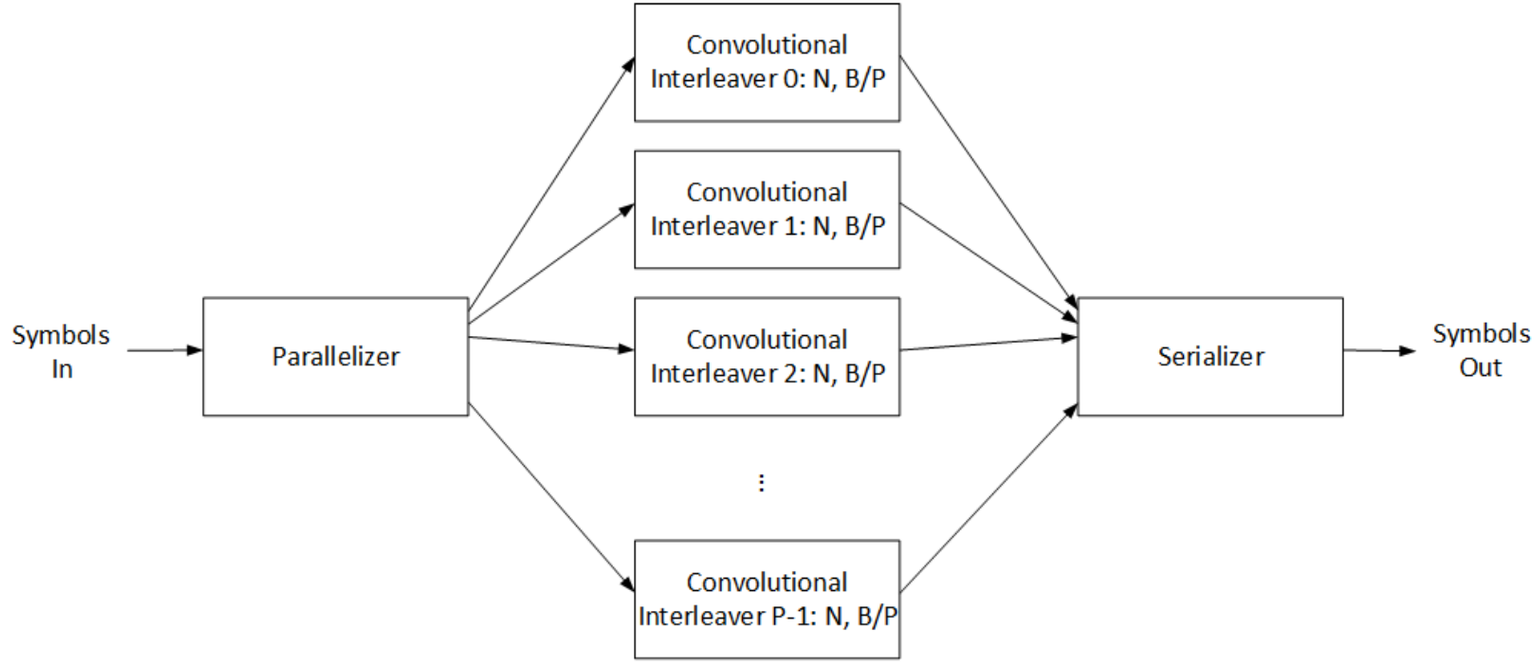
When *N*=4 and *B*=8, the symbols into the interleaver S_0, S_1, S_2, … will produce an output T_i of the following when the initial contents of the interleaver are set to zero:

$T_0, T_1, T_2, T_3, T_4, T_5, \ldots T_{63}, \ldots = S_0, 0, 0, 0, S_4, 0, 0, 0, S_8, 0, 0, 0, S_{12}, 0, 0, 0, S_{16}, 0, 0, 0, S_{20}, 0, 0, 0, S_{24}, 0, 0, 0, S_{28}, 0, 0, 0, S_{32}, S_1, 0, 0, S_{36}, S_5, 0, 0, S_{40}, S_9, 0, 0, S_{44}, S_{13}, 0, 0, S_{48}, S_{17}, 0, 0, S_{52}, S_{21}, 0, 0, S_{56}, S_{25}, 0, 0, S_{60}, S_{29}, 0, 0, \ldots$



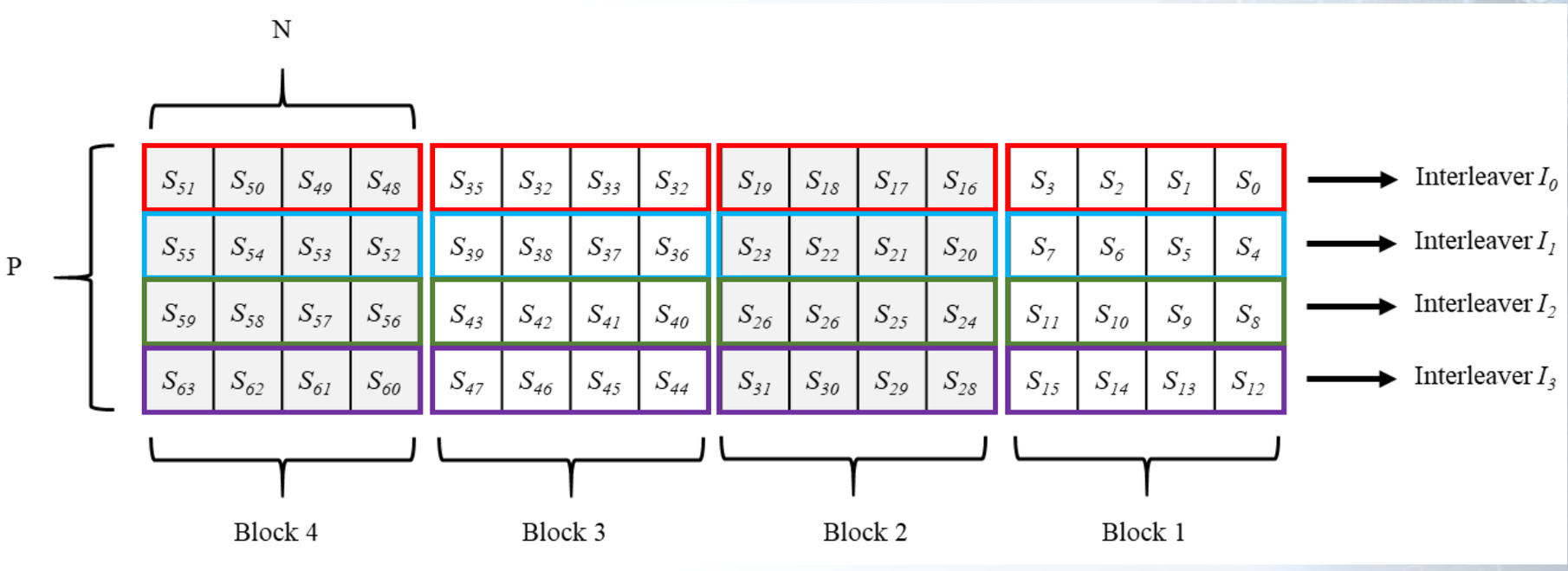
Parallel Convolutional Interleaver

A single large interleaver of size (*N*, *B*) can be shown to be equivalent to *P* smaller interleavers of size (*N*, *B/P*) operating in parallel.



Parallelizer

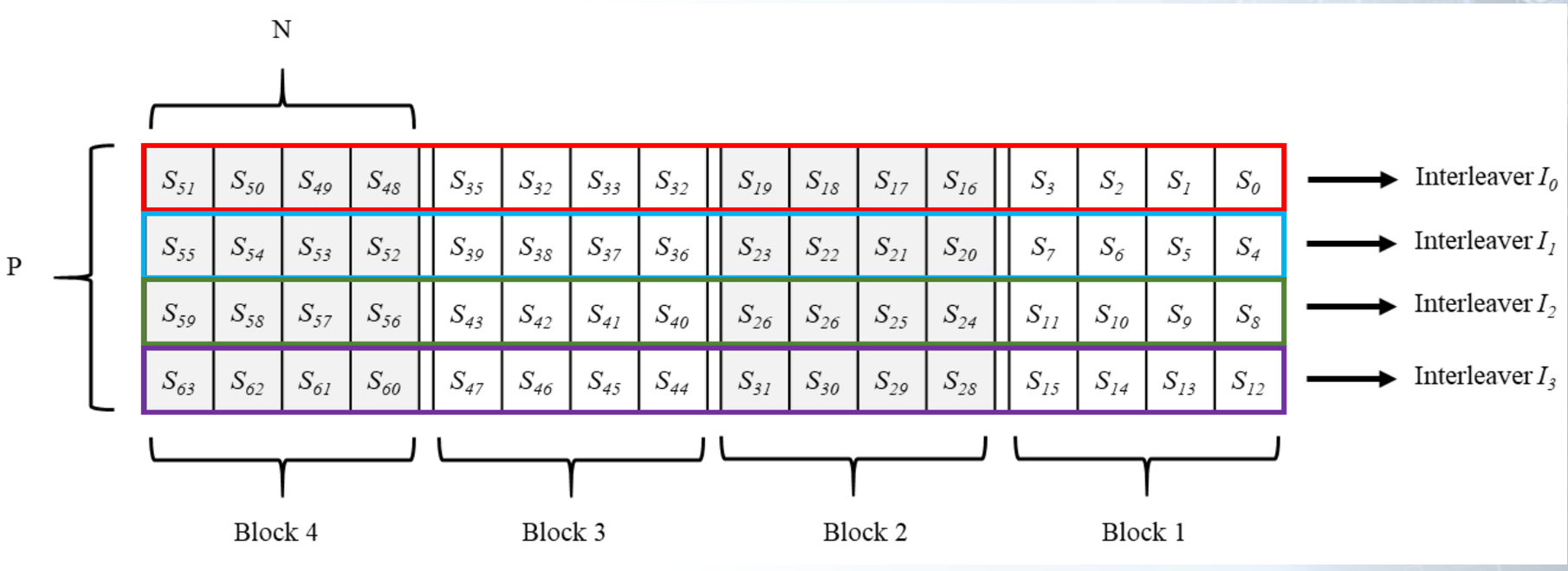
The input data symbols, S_i, are demultiplexed into to queues to form blocks of data that are *P*=4 rows and *N*=4 columns.



Each parallel interleaver (*P*=4) is of size *N*=4 and *B*=2.

Parallelizer

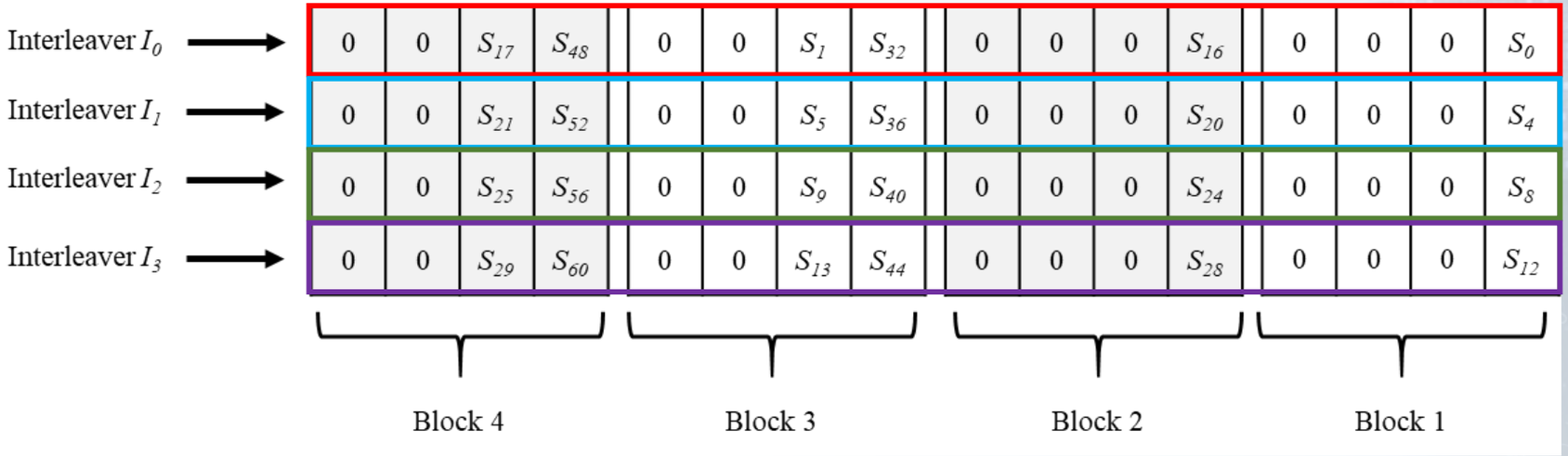
The input data symbols, S_i, are demultiplexed into to queues to form blocks of data that are *P*=4 rows and *N*=4 columns.



Each parallel interleaver (*P*=4) is of size *N*=4 and *B*=2.

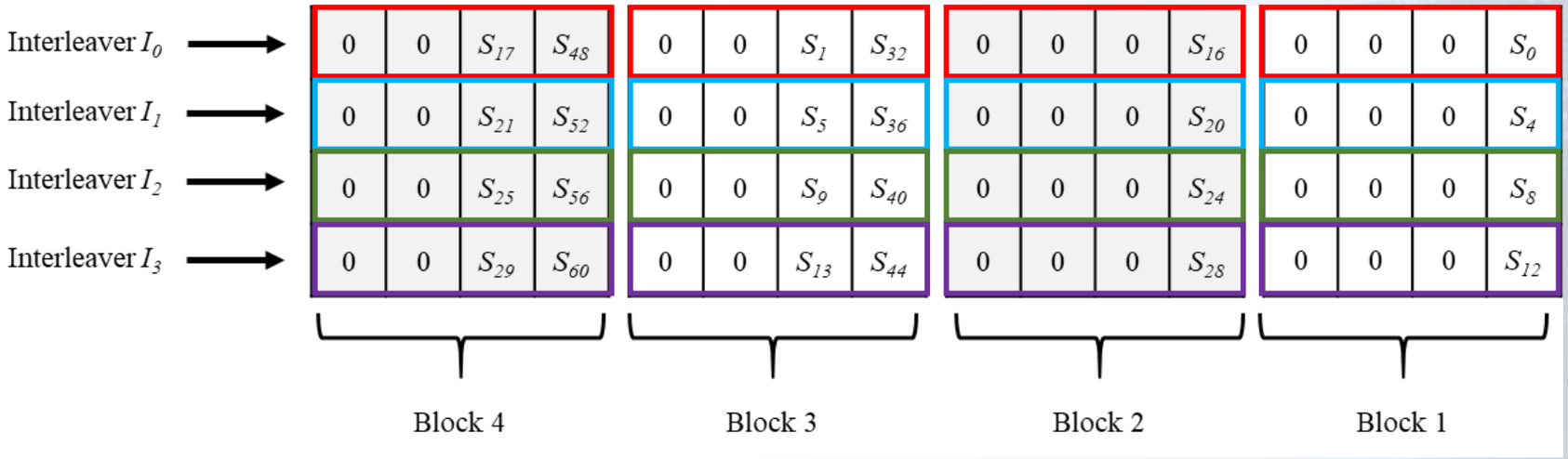
Serializer

Each row stores the output of an interleaver.



Serializer

Each row stores the output of an interleaver.

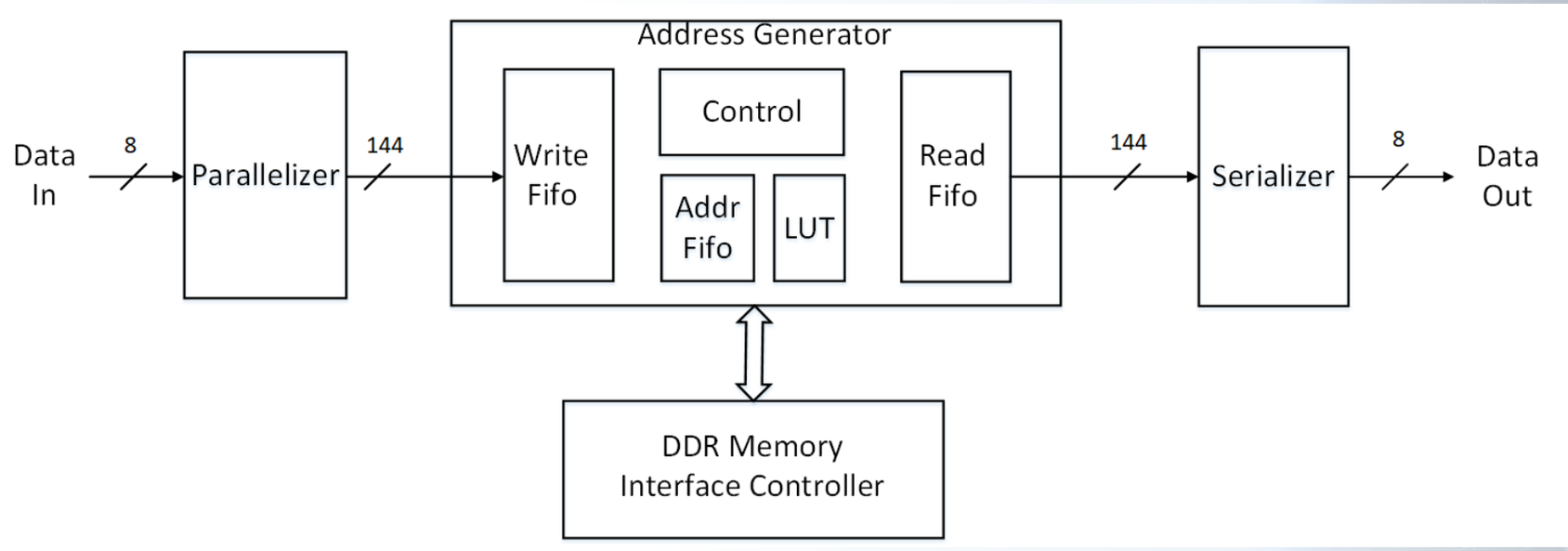


After serialization, the output symbols are in the same order as those of the single large interleaver:

$U_0, U_1, U_2, U_3, U_4, U_5, \ldots U_{63}, \ldots = S_0, 0, 0, 0, S_4, 0, 0, 0, S_8, 0, 0, 0, S_{12}, 0, 0, 0, S_{16}, 0, 0, 0, S_{20}, 0, 0, 0, S_{24}, 0, 0, 0, S_{28}, 0, 0, 0, S_{32}, S_1, 0, 0, S_{36}, S_5, 0, 0, S_{40}, S_9, 0, 0, S_{44}, S_{13}, 0, 0, S_{48}, S_{17}, 0, 0, S_{52}, S_{21}, 0, 0, S_{56}, S_{25}, 0, 0, S_{60}, S_{29}, 0, 0, \ldots$

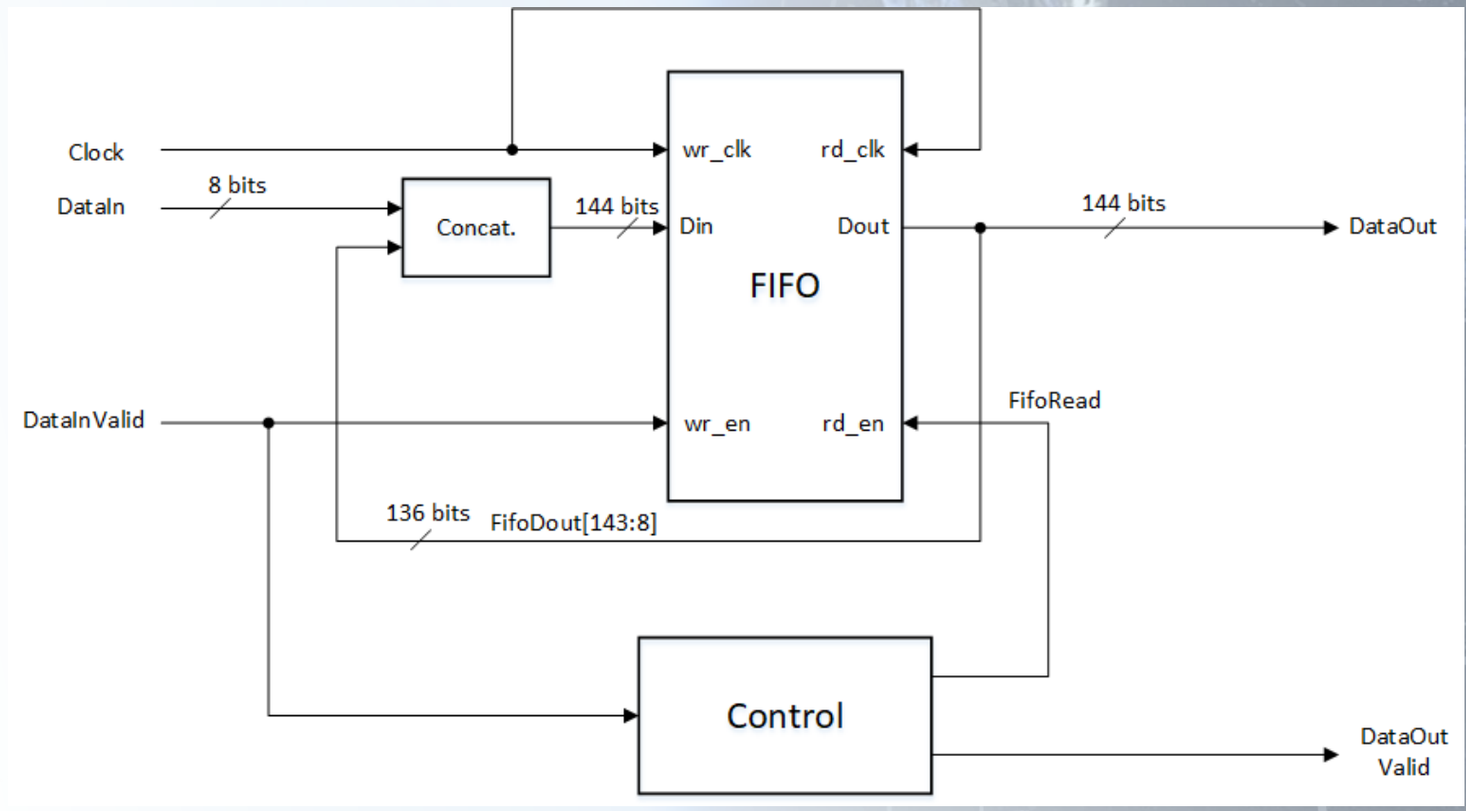
Implementation in FPGA

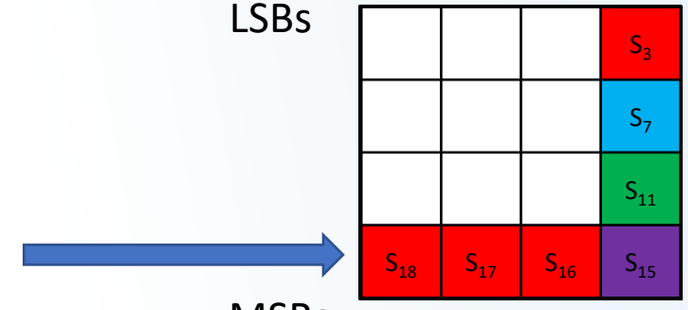
- A parallel-18 (*P*=18) convolutional interleaver was implemented
- Data symbols are represented by 8 bits
- 144 out of 512 bits are used in the DDR interface
- The same position in each of the *P* smaller interleavers is stored at the same DDR address



Parallelizer

- **Each block is stored in a 144 bit wide by *N* deep FIFO**
- **Symbols are written into the most significant bits (bits 143:136) of the FIFO to create the first row of a data block.**
- **When the first row is filled, the FIFO output is shifted by 8 bits and concatenated with the incoming 8-bit symbols.**

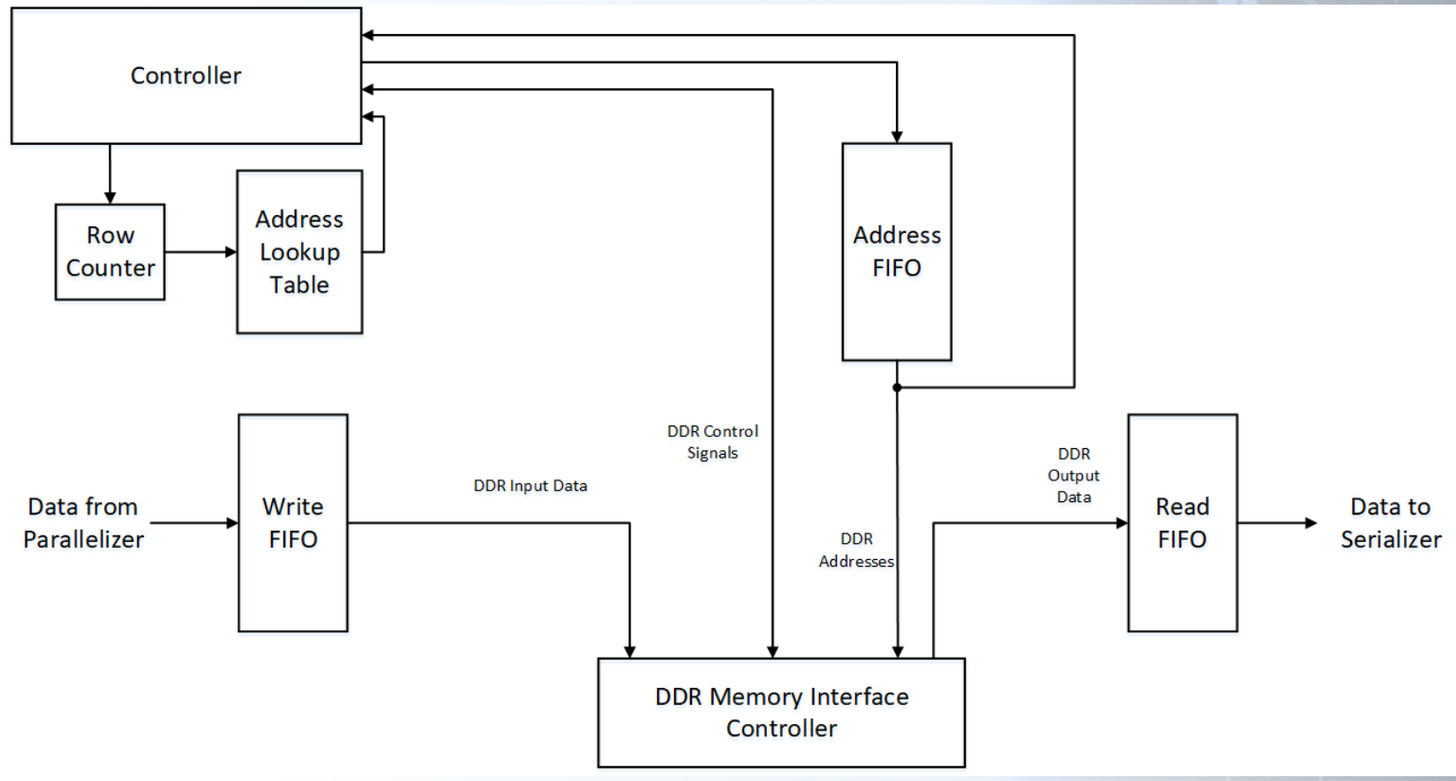




Parallelizer FIFO, N=4, P=4

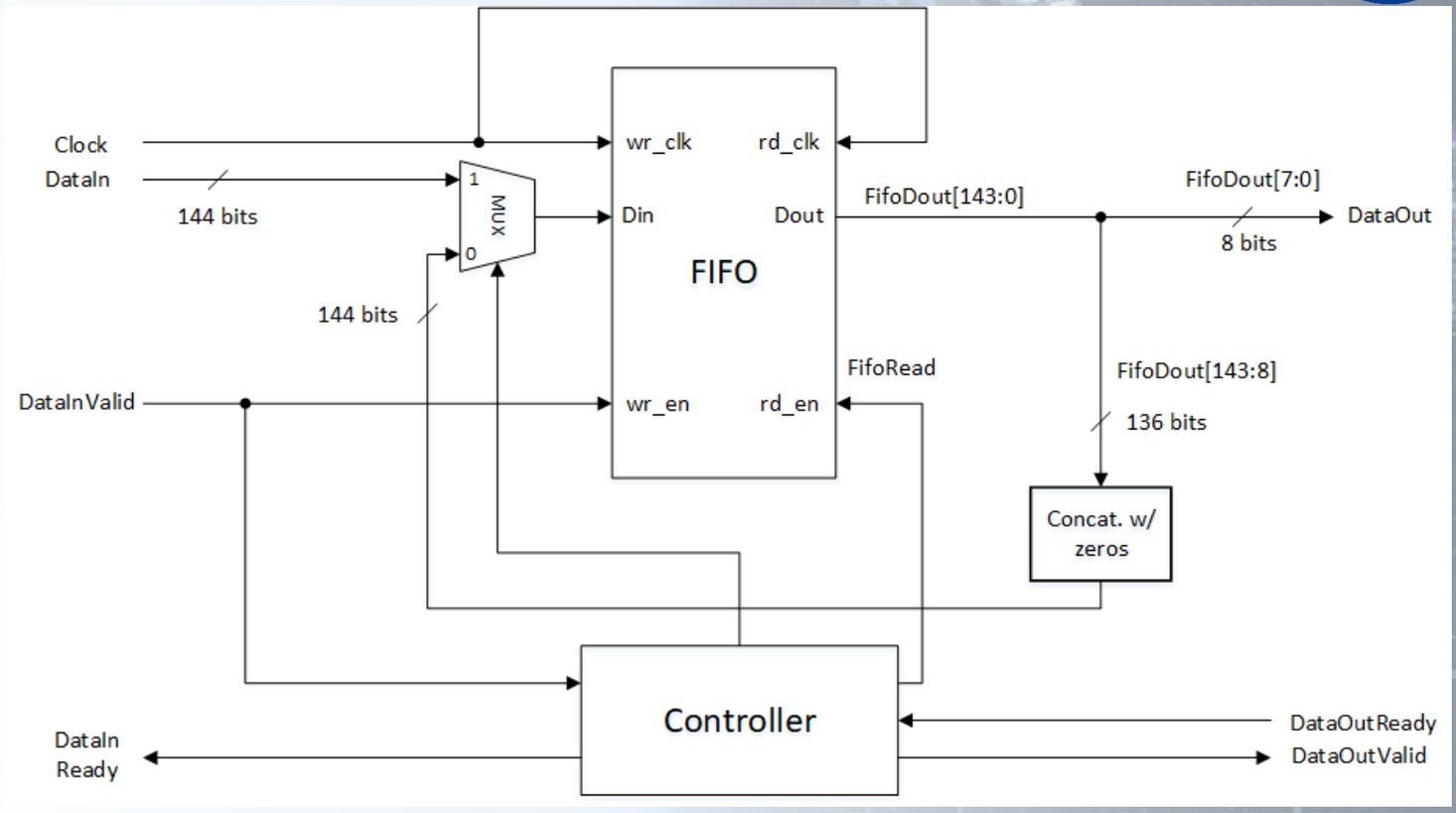
Address generator

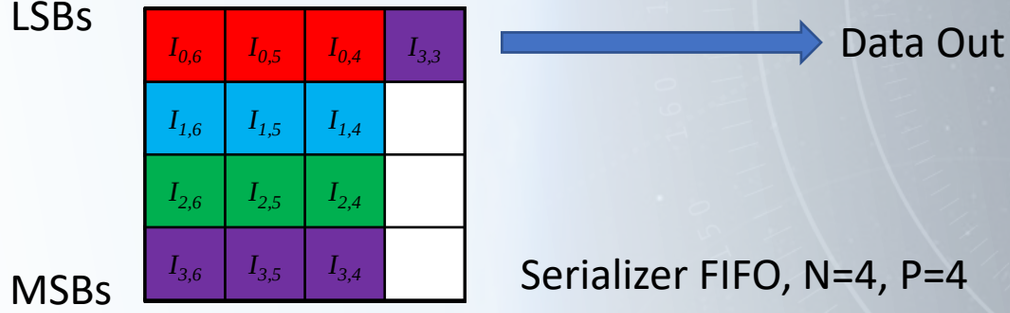
- **Stores data to be written to and read from the DDR**
 - **Write FIFO**
 - **Read FIFO**
- **Generates DDR addresses**
 - **The lookup table contains the starting address of each row and is addressed by the row counter the first time through**
 - **The controller increments the addresses and stores them in the Address FIFO**
 - **Once the address in the row becomes equal to the starting address for the next row, it is set back to the initial address for the row.**



Serializer

- **Accepts a 144-bit vector containing parallel groups of 18 symbols and writes them into the serializer FIFO.**
- **When the FIFO contains a full data block, circular reads are started so that oldest symbol (bits 7:0) of each vector is output from the interleaver.**
- **The remaining FIFO data (bits 143:8) is shifted by one byte and fed back into the FIFO.**

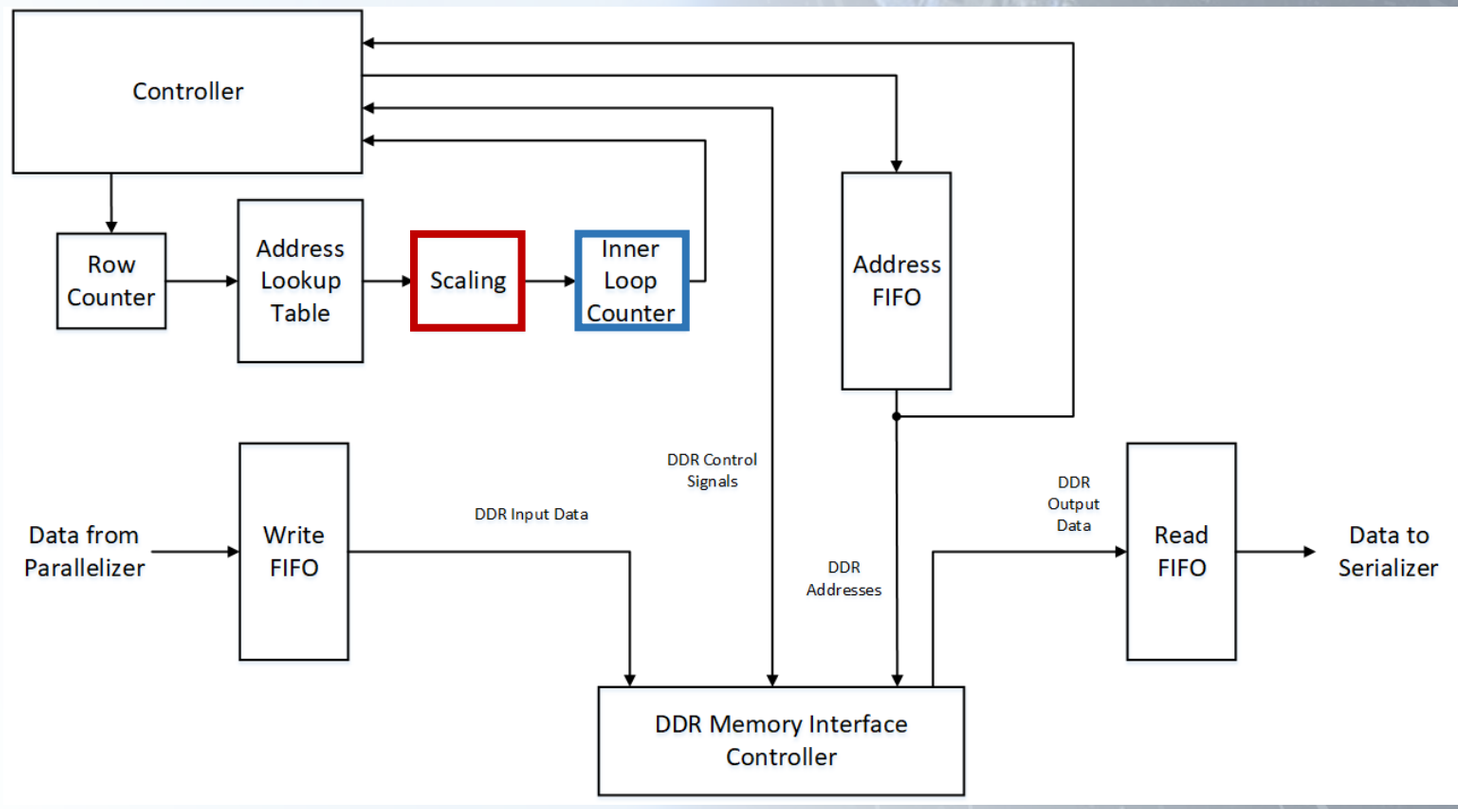




Serializer FIFO, N=4, P=4

Soft Symbol Deinterleaver

- Parallel-9 implementation
- Stores 16 slots, with 3 bits per slot in a memory location (utilizes 432 of 512 bits available in the DDR)
- Inner loop counter enables multiple writes to DDR memory for PPM orders higher than PPM-16
- Scaling factor scales the addresses for higher orders
- DDR4 with 2400 MHz clock rate will implement all CCSDS HPE 2 GHz slot modes



Interleaver Utilization Statistics

- **Interleaver was implemented in VHDL and tested in the FPGA**
- **Interleaver symbol rates: 100 MHz for PPM-16 and 50 MHz PPM-32**

Resource	Parallelizer	Address Generator	Serializer	DDR Memory Interface Controller	Total	% Utilization
Slice LUTs	83	293	234	10,406	11,016	3.6
Slice Registers	304	753	304	6,782	8,143	1.3
Muxes	0	0	0	219	219	0.1
Slices	73	214	97	3,451	3,835	5.1
LUT as Logic	83	293	237	8,936	9,549	3.1
LUT as Memory	0	0	0	1,470	1,470	1.1
BRAM	2	8	2	0	12	1.2

Conclusion

- An algorithm for using *P* parallel convolutional interleavers multiplexed together was designed and has been shown to be equivalent to a large convolutional interleaver.
- This algorithm was implemented in an FPGA and uses external DDR memory.
- The design takes up very little space in the FPGA, leaving significant room for other parts of the design.
- The same code can be used for a soft decision deinterleaver, with changes to the addresses stored in the lookup table and minor changes to the design.



Thank You!

www.nasa.gov/SCaN

